韓國컴퓨터情報學會 論文誌
第14卷 第3號, 2009. 3.

2009-14-3-5-1

소프트웨어 비용산정을 위한 SVM의 파라미터 선정과 응용에 관한 연구

권 기 태*, 이 준 길**

A Study on the Selection of Parameters and Application of SVM for Software Cost Estimation

Kwon Ki-Tae*, Lee Joon Gil**

요 약

소프트웨어 개발 초기 단계에서 소프트웨어 개발비용을 정확하게 예측하는 것은 프로젝트의 성패를 결정짓는 중요한 요소이다. 본 논문에서는 서포트 벡터 머신을 이용하여 소프트웨어 개발비용을 추정하고자 한다. 서포트 벡터 머신은 벡터 공간에서 선형 및 비선형의 경계면을 찾아 입력 데이터를 분류하는 방법으로서 분류 문제에 효과적이다. 하지만 사용자 정의에 의한 파라미터에 의존적이어서 최적의 파라미터를 선택하는 어려움이 있다. 본 연구에서는 서포트 벡터 머신에서 사용하는 파라미터 선택을 위한 개선된 방법을 제안하고, 이러한 최적의 파라미터를 가진 서포트 벡터 머신을 이용하여 소프트웨어 개발비용을 추정하였다. 본 연구의 결과 기존 소프트웨어 비용산정 기법에 비해 향상된 결과를 나타내었다.

Abstract

The accurate estimation of software development cost is important to a successful development in software engineering. This paper presents a software cost estimation method using a support vector machine. Support vector machine is one of the efficient techniques for classification, and it is the classification method of input data based on Maximum-Margin Hyperplane. But SVM has the problem of the selection of optimal parameters, because it is dependent on user's parameters. This paper selects optimized SVM parameters using advanced method, and estimates software development cost. The proposed approach outperform some recent results reported in the literature.

▸ Keyword : 소프트웨어 개발비용(Software Development Cost), 서포트 벡터 머신(Support Vector Machine), 최적 파라미터(Optimal Parameters))

• 제1저자 : 권기태
• 투고일 : 2009. 1. 4, 심사일 : 2009. 2. 2, 게재확정일 : 2009. 3. 23.
* 강릉대학교 컴퓨터공학과 교수 ** 강릉대학교 정보전산원 근무

Ⅰ. 서 론

소프트웨어 개발 초기 단계에서 소프트웨어 개발 비용을 정확하게 예측하는 것은 프로젝트의 성패를 결정짓는 중요한 요소이다. 비용 초과로 고객이 프로젝트를 취소할 수도 있고, 개발 업체는 실제 소요될 비용보다 비용을 적게 예측함으로써 이윤을 남기지 못하고 많은 시간을 소모하게 될 수도 있다. 소프트웨어 생명주기 초기에 개발 비용을 정확히 예측하면, 프로젝트 관리자들은 어떤 자원이 필요한지, 적합한 자원들을 언제 배정해야 할 지를 알 수 있다.

소프트웨어 비용산정 모델에 관한 주요 연구는 1965년 169개 소프트웨어 프로젝트의 104가지의 속성에 관한 SDC의 광범위한 연구로 시작되었다. 이 모델을 기초로 1960년대 후반과 1970년대 초반 부분적으로 유용했던 일부 모델들이 유도되었다. 1970년대 후반에 SLIM, Checkpoint, PRICE-S, SEER, COCOMO 등과 같은 알고리즘 모델이 개발되었다. 이들 비용산정 모델의 개발자 대부분이 동일한 시기에 비용산정 모델을 개발하기 시작했지만, 그들은 모두 유사한 딜레마에 빠졌다. 즉, 소프트웨어의 크기가 커지고 복잡해짐에 따라, 소프트웨어 개발비용을 정확하게 예측하기가 점점 더 어렵다는 것이다. 알고리즘 모델 자체의 문제점과 더불어 매우 빠르게 변화하는 개발 환경의 영향으로 정확도가 높은 알고리즘 모델을 개발하기란 매우 어렵다[1,2].

1980년대에는 알고리즘 모델이 폭넓게 이용되었으며, 이 시기의 모델들은 다양한 규모와 환경적인 데이터 집합을 이용하여 비교하였다. 이러한 연구에서 얻은 주요한 결론은 비용산정 모델들은 환경이 다른 경우에 측정하지 못한 인자들이 적용된다면 성능이 떨어진다는 것이다. 1990년에는 Abdel-Hamid와 Madnick 같은 연구자들은 소프트웨어 개발은 복잡한 동적인 프로세스이며 복잡함과 생산성에서 나타나는 다양성을 설명할 수 있는 변경과 관련된 관계성을 거의 알지 못함을 알게 되었다. 따라서, 1990년대에는 기계 학습 알고리즘에 기반한 비모수 모델링 기법의 소개 및 산정법등이 등장하였다[3].

기계학습에 의한 소프트웨어 개발비용 예측 방법으로 가장 먼저 사용된 기법은 신경망에 의한 비용예측이다. 이어서 사례기반 추론 방법을 도입하여 소프트웨어 개발비를 예측하였고, 또한 트리 기반 방법으로 회귀트리, 의사결정트리 등을 활용하여 소프트웨어 비용예측을 시도한 연구들도 수행되었다. 신경망, 사례기반추론 또는 회귀모형을 이용한 소프트웨어 비용예측의 정확도를 비교하면 연구자에 따라 비용예측의 정확도가 다소 차이가 있는 것으로 보고되고 있다[4,5,6].

본 연구에서는 기계학습 분야에서 새롭게 주목을 받고 있는 서포트 벡터 머신(SVM)을 소프트웨어 개발비 예측에 적용하여 그 정확도를 기존의 방법과 비교하고자 한다. Vapnik에 의해 제안된 SVM은 소수의 데이터 집합으로 훈련시켜도 일반화 능력이 뛰어나고, 이용되는 데이터 집합의 확률분포를 사전에 가정하지 않는 장점을 가지므로[7], 패턴인식과 분류 문제에 있어서 좋은 성과를 나타내어 생체정보학, 주가 예측, 침입탐지 등 활용 범위를 점차 넓혀가고 있다. SVM은 고차원 데이터 집합에 잘 동작하기 때문에 프로젝트 비용산정과 같이 영향을 주는 요소가 다양한 데이터 중심의 문제에 적합하다. SVM은 아주 강력한 분류기로써, 올바른 인자를 얻으면 어떤 분류 기법보다 더 정확하고 더 잘 동작 한다. 하지만, 한 가지 단점은 최적 커널 변환 함수와 이 함수의 인자들이 모든 데이터 집합마다 약간씩 달라 매번 이들을 찾아야 한다는 것이다. 이에 본 논문에서는 개선된 파라미터 선택 방법을 적용하여 SVM에서 사용하는 사용자 정의 파라미터들의 최적값을 예측하고 이를 이용하여 소프트웨어 개발비용을 추정하고자 한다.

본 논문의 2장에서는 관련 연구에 대해 살펴보고, 3장에서는 개선된 GSS 방법을 적용한 최적 파라미터 결정에 대하여 기술한다. 4장에서는 제안된 방법을 적용한 실험 및 분석 결과를 설명하고, 5장에서 결론과 향후 연구 방향에 대해 알아본다.

Ⅱ. 관련 연구

많은 소프트웨어 비용산정 알고리즘 모델이 지난 30여년 이상 개발되어 왔지만, 알고리즘 모델 자체에 근본적인 한계를 가지고 있으므로 기계학습 알고리즘에 기반한 비모수 기법에 대한 관심이 제기되고 있다.

2.1 서포트 벡터 머신

VM은 Vapnik에 의해 제안된 통계적 학습이론으로 두 범주를 갖는 객체들을 분류하는 방법이다[8]. SVM은 분류 오류 확률을 최소화 하는 구조적 위험 최소화(structural risk minimization) 방법에 기초하고 있다. SVM 은 인공신경망과 비슷한 수준의 높은 예측력을 나타낼 뿐만 아니라 인공신경망의 한계점으로 지적되었던 과대적합, 국소최적화와 같은 한계점들을 완화하는 장점을 갖고 있다.

SVM은 기본적으로 학습 데이터를 분류하고 그 분류할 때

사용하는 결정함수에 의해 발생하는 일반화 오차(generalization error)를 최소로 하는 결정함수를 찾는 것이다.

주어진 학습 데이터 $(x^1,y^1),...,(x^l,y^l)$, $x\in R$, $y\in\{-1,+1\}$를 분류하는 결정함수를 D라 하면 결정함수 D는 식(1)과 같다.

$$D(X) : W^T \bullet X+b=0 \quad \cdots\cdots (1)$$

결정함수를 초평면(hyperplane)이라 하는데 SVM에서는 최종적으로 학습 데이터를 두 그룹의 데이터로 분리할 수 있는데, 이때 최적분리 경계면과 각 그룹의 가장 근접한 데이터를 SV(Support Vector)라 하며, 각 그룹의 SV간의 거리 $\frac{2}{\|w\|}$가 최대가 되는 지점에서 최적 분리 경계면이 설정된다.

그러나 일반적으로 실세계에서는 선형분리가 가능하지 않으며, 이러한 경우에 결정된 최적 분리 경계면은 높은 분류능력을 갖지 못한다. 선형분류가 어려운 경우에 SVM은 커널함수라는 매핑함수를 이용하여 고차원 특징공간을 선형분류 가능한 공간으로 매핑하여 분류능력을 일반화시킨다.

2.2 SVM Regression

SVM 분류를 회귀 문제에 적용하여 훈련 데이터에 의존한 SVM Regression 예측 모델을 만들 수 있다. SVM Regression은 SVM Classification과 같은 방식으로 구하며 그 절차는 (그림 1)과 같다[9,10]

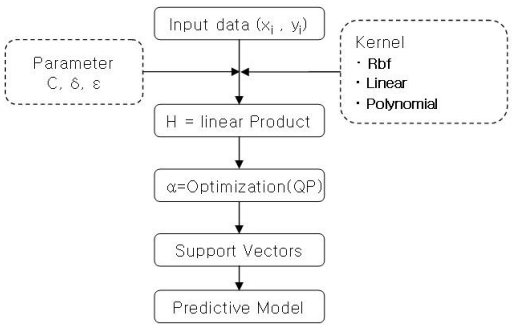

그림 1. SVM 처리 순서도
Fig. 1. Flowchart of SVM

SVM Regression은 학습 데이터 $D=\{(x_i,y_i)\in R^n\times R,\ i=1,2,...,l\}$ $x\in R^n$, $y\in R$가 있을 때 선형회귀 초평면은 식(2)와 같다.

$$f(x,w) = W^T X+b \quad \cdots\cdots (2)$$

SVM Regression에서는 분류의 마진 대신에 근사값 오류를 측정한다. 이 ε-insensitivity zone을 갖는 오류함수는 식(3), (그림 2)와 같다.

$$E(x,y,f) = |y-f(x,w)|_\epsilon = \begin{cases} 0 & \text{if } |y-f(x,w)|\le\epsilon \\ |y-f(x,w)|-\epsilon, & otherwise \end{cases} \quad \cdots\cdots (3)$$

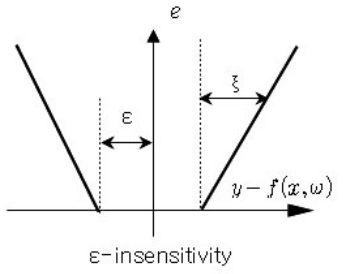

그림 2. ε-insensitivity 함수
Fig. 2. ε-insensitivity Function

(그림 3)은 SVM Regression에서 사용되는 파라미터들, 실데이터, 예측 데이터들 사이의 관계를 보여준다. (그림 3)에서 검은 사각형은 실제값이며 Support Vector를 나타낸다.

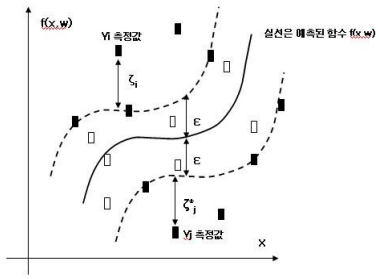

그림 3. SVM에서 파라미터 관계
Fig. 3. Parameter Relation of SVM

식(6)의 실험적 오류를 식(4)와 같이 나타내고, 식(4)와 $\|W\|^2$을 동시에 최소로 함으로써 식(2)의 초평면을 구할 수 있다. 식(4)를 최소로 하는 것은 식(5), 식(6), 식(7)을 조건으로 식(5)를 최소화하는 것과 같다.

$$R_{emp}^{e}(w,b)=\frac{1}{l}\sum_{i=1}^{l}\left|y_i-W^Tx_i-b\right|_e \quad \cdots\cdots (4)$$

$$R_{w,\xi,\xi^*}=\left[\frac{1}{2}\parallel w\parallel^2+C\left(\sum_{i=1}^{l}\xi_i+\sum_{i=1}^{l}\xi_i^*\right)\right] \quad \cdots\cdots (5)$$

$$y_i-W^Tx_i-b\le\epsilon+\xi_i,\ i=1,2,...,l \quad \cdots\cdots (6)$$

$$W^Tx_i+b-y_i\le\epsilon+\xi_i^*,\ i=1,2,...,l \quad \cdots\cdots (7)$$

$$\xi_i\ge 0,\xi_i^*\ge 0,\ i=1,2,...,l \quad \cdots\cdots (8)$$

이후 SVM Classification과 같이 최적화 문제를 풀면 식(10), 식(11) 조건하에 회귀식(9)를 구할 수 있다.

$$f(x)=\sum_{i=1}^{n}(\alpha_i-\alpha_i^{'})K(x_i,x)+b \quad \cdots\cdots (9)$$

$$\sum_{i=1}^{n}(\alpha_i-\alpha_i^{'})=0 \quad \cdots\cdots (10)$$

$$0\le\alpha_i,\alpha_i^{'}\le C,\ i=1,...,n \quad \cdots\cdots (11)$$

III. 본 론

소프트웨어 공학에서 SVM을 이용한 연구는 제한적인데, 단순히 SVM에서 사용되는 파라미터들을 사용자의 직관으로 가정하여 소프트웨어 개발비용을 추정하거나 소프트웨어 신뢰성을 예측하였다[11,12,13,14].

본 논문에서는 파라미터 결정 방법을 사용자의 직관적인 지식으로 하는 방법을 개선하여 소프트웨어 개발비용 산정의 정확도를 결정하는 척도인 MMRE를 최소화하는 파라미터 값을 결정하고 다른 파라미터들을 그 점근적 관계성에 따라 계산하는 개선된 방법을 이용하여 최적의 파라미터 조합을 갖는 비용산정 모델을 구한다.

3.1 파라미터간의 관계

V파라미터의 선택은 비용산정의 정확성을 향상시키고, 실험영역 이해와 산정 시간의 단축을 위하여 매우 중요하다.

C는 학습 데이터의 결과 범위와 관련이 있고 이상치 데이터에 민감하다. 일반적으로 식(12)와 같은 범위에서 C의 값을 정한다.

$$C=\max\left(\left|\bar{y}+3\sigma_y\right|,\left|\bar{y}-3\sigma_y\right|\right) \quad (12)$$

여기서 $\bar{y}$: 학습 데이터의 평균

σ_y : 학습 데이터의 표준편차

ε는 학습 데이터의 잔차들의 분산과 관련이 있으며 실험적 결과로서 식(13)의 관계가 있음이 알려졌다.

$$\epsilon\propto\sigma\sqrt{\frac{\ln n}{n}} \quad \cdots\cdots (13)$$

[13]에 의하면 σ와 C는 (그림 4)와 같은 관계가 있다.

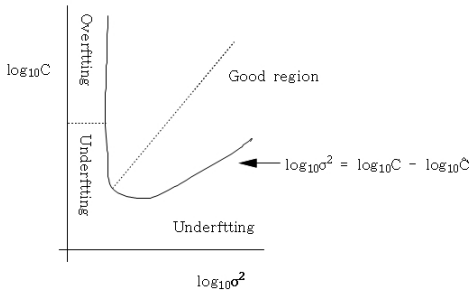

그림 4. σ와 C의 관계
Fig. 4. Relation of σ and C

3.2 개선된 파라미터 선택 방안

SVM은 분류 최적화에 대한 일반화 능력이 뛰어난 반면에 데이터 집합에 따라 적합한 커널 변환 함수의 파라미터 값이 달라 매번 이들을 찾아야 하는 문제가 있다. 사용자 정의에 의존하는 파라미터 값은 SVM의 성능에 있어서 매우 중요한 역할을 하게 된다. 정확한 비용추정을 위하여 사용자는 다양한 변경값을 파라미터에 적용하게 되는데, SVM은 신경망과 같은 블랙박스 기법을 사용하고 있어서, 파라미터의 증가 감소에 따른 결과를 미리 예측하기가 어렵다.

SVM을 이용한 소프트웨어 비용산정은 학습 데이터에 포함된 입력 특성들을 사용하여 개발비용을 추정하는 선형 또는 비선형 SVM 모델을 만들고, 만들어진 SVM 모델을 이용하여 테스트 데이터의 비용산정에 활용하는 방법이다. 입력 특성으로 사용되는 비용산정 영향 요소로는 개발 팀 규모, 기능점수 등 비용산정에 영향을 미칠 가능성이 있는 다양한 자료들이 수치화되어 사용되어진다.

SVM을 사용하는 소프트웨어 비용산정 모델의 이슈는 적합한 파라미터 값 선택에 있다. SVM을 이용한 기존의 소프트웨어 개발비용 산정 방법은 임의의 파라미터 값을 무작위로 선택하거나[11,12] 입력 데이터를 이용하여 기준 파라미터 집합을 구성하고 파라미터 집합을 일정한 방향으로 증가시켜

추정하는 방법을 취하였다[13]. 이는 고정된 특정위치로 고정되거나 일정 방향으로만 변경되기 때문에 SVM의 회귀식이 최적화된 파라미터를 가지는 것이 아니라 지형적인 국소값의 한계를 가질 수 있다.

본 논문에서 파라미터를 선택하고 비용 산정을 계속하는 방법은, 작업공간에서 σ를 선택한 후 이를 기준으로 (그림 4)에 나타나 있는 파라미터 관계식을 이용해 C, ε를 계산하고 이 파라미터로서 입력 데이터에 대한 결과값 추정 후 MMRE를 계산한다. (그림 5)와 같이 계산된 MMRE를 이전 값과 비교하여 MMRE 값이 좋으면 파라미터 값을 증가시키고, 나쁘면 파라미터 값을 감소시켜 추정을 반복한다.

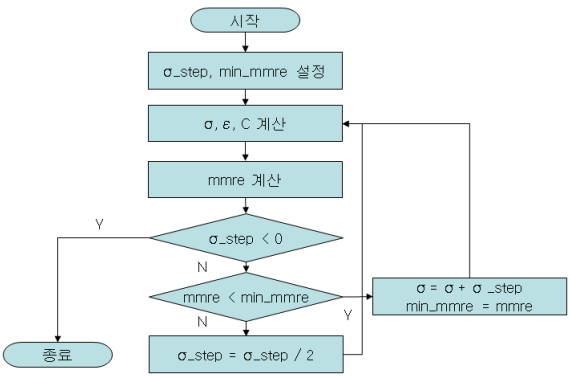

그림 5. 파라미터 결정 과정
Fig. 5. Selection of Parameters

IV. 실험 및 분석

본 논문에서 사용된 데이터는 ISBSG(International Software Benchmarking Standards Group)에서 수집한 데이터[15]를 사용했으며, 이 중 테스트에 사용한 데이터는 데이터 신뢰성, 기능점수, 최대 팀 크기를 고려하고, 이상치를 제거하여 209개를 선정하여 테스트 데이터로 사용하였다. 이 중 150개를 학습 데이터로 사용하여 비용 산정 모델을 생성하고 나머지 59개로 생성된 모델을 검증하였다.

SVM을 이용하여 소프트웨어 개발 비용을 추정한 결과가 〈표 1〉과 같고, 추정 결과인 MMRE의 분포도를 보면 (그림 6)과 같다.

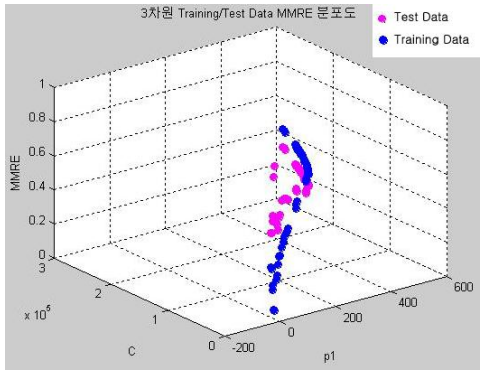

그림 6. 학습/테스트 데이터 MMRE 분포도
Fig. 6. MMRE of Learning/Testing data

〈표 1〉의 각각의 파라미터 C, σ, ε값을 가지고 학습 데이터를 실험하였고, 이 값으로 테스트 데이터를 각각 실험하였다. 또 학습 데이터와 테스트 데이터의 평가척도로는 소프트웨어 개발비용 산정 모델의 평가 척도인 MMRE와 PRED 값을 계산하여 비교하였다.

표 1. SVM을 이용한 추정된 결과
Table 1. Estimated Results using SVM

	C	σ	ε	Training Data		TestData	
				MMRE	PRED(25)	MMRE	PRED(25)
1	749110	859.72	103.55	0.59	0.37	0.36	0.36
2	3774510	1940.23	233.69	0.58	0.43	0.36	0.41
3	1705584	1302.15	156.84	0.59	0.38	0.35	0.37
4	748742	859.50	103.52	0.59	0.37	0.36	0.36
5	91078	284.74	34.30	0.64	0.39	0.47	0.37
6	804648	891.43	107.37	0.59	0.37	0.36	0.37
7	7789429	2789.16	335.94	0.62	0.41	0.39	0.44
8	4059866	2012.28	242.37	0.59	0.43	0.37	0.41
9	57742	218.50	26.32	0.68	0.44	0.53	0.39
10	98003	286.36	34.49	0.64	0.39	0.47	0.37
11	4007301	1999.33	240.81	0.59	0.43	0.37	0.41
12	190712	425.10	51.20	0.61	0.39	0.40	0.37
13	10185973	3189.98	384.22	0.64	0.39	0.40	0.39
14	4811307	2191.19	263.92	0.61	0.42	0.38	0.44
15	736868	852.62	102.69	0.59	0.37	0.36	0.36
16	11249991	3352.61	403.81	0.65	0.40	0.40	0.39
17	8511397	2915.72	351.19	0.63	0.41	0.39	0.42
18	7298860	2699.44	325.14	0.62	0.41	0.39	0.44
19	1809780	1341.56	161.59	0.59	0.39	0.35	0.37
20	745107	857.38	103.27	0.59	0.37	0.36	0.36

〈표 1〉을 보면 학습 데이터 추정은 MMRE가 58%부터 65%까지, PRED가 37%부터 44%까지 값을 보이고 있으며, 테스트 데이터 추정은 MMRE가 35%부터 47%까지, PRED는 36%부터 44%까지 결과를 보이고 있다. 〈표 1〉에서 학습 데이터는 C=3774510, σ = 1940.23, ε= 233.69에서 가장 좋은 결과를 보이고 있으며, 테스트 데이터는 C=1705584, σ = 1940.23, ε= 156.84에서 가장 좋

은 결과를 보이고 있다.

특히 위의 산정 결과를 보면 학습 데이터와 테스트 데이터의 일부 데이터를 제외하고 평균적인 결과를 보이고 있어 매우 안정적인 결과를 보이고 있다. 또한 테스트 데이터와 학습 데이터가 모두 비슷한 추정 결과를 보이고 있어 본 논문에서 제안한 비용 산정 모델이 유용한 모델이라 할 수 있다. (그림 7)은 테스트 데이터의 실제값과 추정값 59개에 대한 비교를 보여 준다.

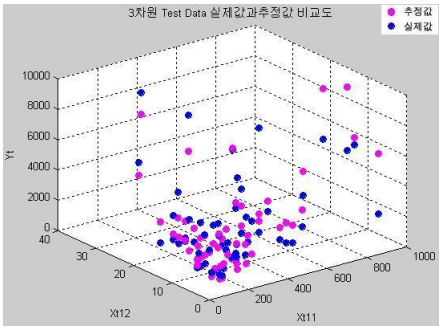

그림 7. 테스트 데이터 실제값과 추정값 비교
Fig. 7. Comparison of Actual Values and Estimated Values

본 논문에서 제안한 SVM 모델과 회귀분석을 통해 유도된 전통적인 표준회귀 모델과 ISBSG 모델[15]은 각각 식(14), 식(15), 식(16)과 같으며, 〈표 2〉는 이 세 모델을 대상으로 테스트 데이터를 적용하여 추정한 결과를 비교한 표이다. 〈표 2〉를 보면 본 연구에서 제안하는 모델의 MMRE와 PRED가 기존 표준회귀 모델과 ISBSG 모델보다 향상된 추정 결과를 보이고 있다. (그림 8)은 모델별 평가척도를 비교한 그래프이다.

표 2. SVM 모델과 전통적 모델의 비교
Table 2. SVM and Traditional Model

	MMRE	PRED(25)
표준회귀 모델	0.52	0.33
ISBSG 모델	0.40	0.40
SVM 모델	0.38	0.44

$$y = \sum_{i=1}^{n} (\alpha_i - \alpha_i^{'}) K(x_i, x) \quad \cdots\cdots (14)$$

$$y = -470.76 + 5.55x_1 + 264.149x_2 \quad \cdots\cdots (15)$$

$$y = 52.48 FP^{0.426} TS^{0.783} \quad \cdots\cdots (16)$$

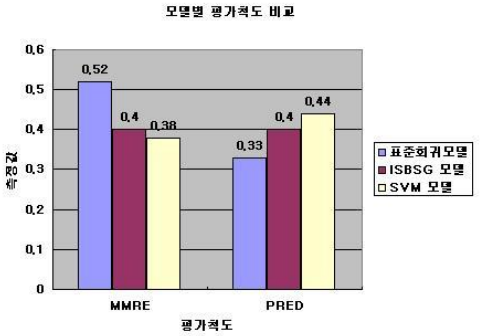

그림 8. 모델별 평가척도 비교도
Fig. 8. Evaluation Metrics of Models

V. 결 론

소프트웨어 개발 초기 단계에서 소프트웨어 개발 비용을 정확하게 예측하는 것은 프로젝트의 성패를 결정짓는 중요한 요소이다. 본 논문의 목적은 입력 데이터와 결과값을 매핑하는 유일하고 정확한 값을 구하는 것이 아니고 소프트웨어 개발비용 추정을 위한 최적화된 SVM 회귀식을 찾는 것이다. 본 논문에서는 SVM을 이용한 소프트웨어 개발비용을 추정을 위해, 먼저 소프트웨어 개발 비용에 영향을 주는 기능점수와 팀 크기를 기반으로 SVM 모델을 위한 최적화된 파라미터를 선택하고 이를 이용하여 개발비용을 추정하였다. 또 ISBSG 데이터를 이용하여 기존 모델들과 제안된 방안으로 추정한 결과를 비교하였다.

본 논문의 학습 데이터와 테스트 데이터를 이용하여 실험한 결과로 볼 때 SVM을 이용한 소프트웨어 개발비용 추정을 위한 파라미터 선택과 응용은 유용한 방안임을 알 수 있으며, 기존 모델들 간의 비교에서도 향상된 결과를 보이고 있다.

향후에는 최적 파라미터 선택을 위해 이미지 처리 알고리즘[16] 및 위치추정 알고리즘[17]을 적용하고, SVM을 이용한 모델을 유지보수 데이터에 적용하는 것뿐만 아니라 소프트웨어 품질관리 분야에도 적용하는 연구도 의미 있는 차후 과제라 할 수 있다.

참고문헌

[1] Linda M. Laird and M. Carol Brenman., "*Software Measurement and Estimation*" Wiley-Interscience, 2006.

〔2〕 Barry W. Boehm et al., "*Software Cost Estimation with COCOMO II*" Prentice-Hall, 2000.
〔3〕 M. A. Parthasarathy, "*Practical Software Estimation*," Addison Wesley, 2007.
〔4〕 Mukhopadhyay et al., "Examining the Feasibility of a Case-based Reasoning Model for Software Effort Estimation," MIS Quarterly, 16, pp. 155-171, June 1992.
〔5〕 Martin Shepperd and Chris Schofield, "Estimating Software Project Effort Using Analogies," IEEE Trans. Software Eng., Vol. 23. No. 12, pp. 736-743, Nov. 1997.
〔6〕 M. Shin and A.L. Goel, "Empirical Data Modeling in Software Engineering using Radial Basis Functions," IEEE Trans. Software Eng. Vol. 26, No. 6, pp. 567-576, June 2000.
〔7〕 K. V. Kumar et al., "Software Development Cost Estimation using Wavelet Neural Networks," The Journal of Systems and Softwares Vol. 81, Issue 11, pp. 1853-1867, Nov. 2008.
〔8〕 Pang-Ning Tan et al., "*Introduction to Data Mining*," Addison Wesley, 2006.
〔9〕 Changha Hwang., "Support Vector Median Regression," Data and Information Science, Vol. 14, No. 1, pp. 67-74, 2003.
〔10〕 Toby Segaran, "*Programming Collective Intelligence*," O'relly, 2007.
〔11〕 Adriano L.I. Oliveira, "Estimation of Software Project Effort with Support Vector Regression," Neuro Computing, Vol. 69, Issue 8, pp. 1749-1753, March 2006.
〔12〕 박찬규, "Support Vector Regression을 이용한 소프트웨어 개발비 예측," 경영과학, 제23권, 제2호, 75-91쪽, 2006년 11월.
〔13〕 Hojung Lim and Amrit L. Goel, "Support Vector Machines for Data Modeling with Software Engineering Applications," in Springer Handbook of Engineering Statistics, Springer, pp. 1023-1037, 2006.
〔14〕 Ping-Feug Dai and Wei-Cahiang Hong, "Software Reliability Forecasting by Support Vector Machines with Simulated Annealing Algorithms," The Journal of Systems and Softwares Vol. 79, Issue 6, pp. 747-755, June 2006.
〔15〕 ISBSG, "Practical Project Estimation : A Tool Kit for Estimating Software Development Effort and Duration," International Software Benchmarking Standards Group, 2001.
〔16〕 최미영, 김계영, 최형일, "자연 영상에서의 정확한 문자 검출에 관한 연구," 한국컴퓨터정보학회 논문지, 제13권, 제5호, 77-84쪽, 2008년 9월.
〔17〕 이관형, 송우영, '레이더 시스템에서 목표물 위치추정 알고리즘에 관한 연구," 한국컴퓨터정보학회 논문지, 제13권, 제5호, 111-116쪽 2008년 9월.
〔18〕 Tron Foss., "A Simulation Study of the Model Evaluation Criterion MMRE," IEEE Trans. Software Eng. Vol. 29, No. 11, pp. 985-995, Nov. 2003.

부 록

소프트웨어 비용산정 모델의 정확도 평가 척도

(1) 상대 오차의 절대값 평균

$$(Mean)Magnitude\ of\ \ RE : (MMRE = \overline{MRE})$$

$$MRE = |\,RE\,|,\ MMRE = \overline{MRE} = \frac{1}{n}\sum_{i=1}^{n} MRE_i$$

상대오차의 절대값인 MRE가 작을수록 모델의 정확도는 높아진다. 반대로 MRE가 클수록 모델의 정확도는 낮아진다.

$MMRE$가 대부분의 비용산정 연구에서 비용산정 모델의 정확도를 평가하는 기준으로 사용되는 이유로는 상대 오차가 큰 양수는 상대 오차가 큰 음수에 의해 상쇄되지 못한다는 장점을 지니고 있기 때문이다.

(2) prediction at level l : $PRED(l)$

$$PRED(l) = \frac{k}{n}$$

위 식에서 n은 전체 데이터의 개수이며 k는 $MRE \leq l$에 포함되는 데이터의 개수이다. 예를 들어 $PRED(25) = 0.83$의 의미는 예측값이 오차 범위 25%안에 드는 데이터의 개수가 전체의 83%에 해당된다는 것이다. PRED는 일반적으로 예측 모델의 정확도를 평가하기에 적절

한 척도이다[18].

저 자 소 개

권 기 태

1986년 서울대학교 계산통계학과 학사
1988년 서울대학교 계산통계학과 석사
1993년 서울대학교 계산통계학과 박사
1996년 Univ. of Southern California, Post-Doc.
1990.9 ~ 현재 강릉대학교 컴퓨터공학과 교수
관심분야 : 소프트웨어 비용산정, 소프트웨어 메트릭스, 소프트웨어 아키텍처

이 준 길

1987년 서울시립대학교 전산통계학과 학사
2000년 강릉대학교 산업대학원 컴퓨터 과학과 석사
2004년 ~ 현재 강릉대학교 컴퓨터공학과 박사과정
현재 강릉대학교 정보전산원 근무
관심분야 : 소프트웨어 비용산정, 소프트웨어 메트릭스, 소프트웨어 아키텍처